

## i Prófleiðbeiningar

Einu leyfilegu hjálpargögnin eru eitt A4 blað (skrifað báðum megin), sem nemandinn hefur búið til.

Ef galli á prófinu kemur í ljós, verður tekið tillit til þess við yfirferð prófsins.

Ef nemandi er óviss um hvort skilningur hans á forritunarspurningu er réttur, getur hann tilgreint hver sá skilningur er í svarreit í opnum spurningum eða sent póst til kennara eftir prófið

## i Leiðbeiningar krossar

Svarið eftirfarandi krossaspurningum með því að merkja við eitt svar í hverri spurningu. Ef ykkur finnst fleiri en eitt svar koma til greina skuluð þið velja svarið sem ykkur finnst „réttast“. Ekki er dregið niður aukalega fyrir rangt svar.

### 1 whileKross

Ef k er 100 og u er 4 hvað er þá prentað út?

```
int k = Integer.parseInt(args[0]);
int u = Integer.parseInt(args[1]);
int t = 1;
while (k > u) {
    k = k / t;
    u = u * t++;
}
System.out.println("k=" + k + " u=" + u);
```

Veldu eitt af eftirtöldu

k=4.1666 u=48

k=12 u=16

k=16.6666 u=24

k=16 u=24

k=50 u=8

---

Maximum marks: 4

## 2 Lesa úr skrá

Lesa á línu úr skránni StringData.txt með eftifarandi setningu:

```
String lina = s.readLine();
```

Hvernig á að skilgreina og smíða hlut sem er geymdur í breytunni s?

**Veljið eitt af eftirtöldu**

- String s = new Scanner ("StringData.txt");
- In s = new In ("StringData.txt");
- String s = new String("StringData.txt");
- In s = In ("StringData.txt");
- Scanner s = new Scanner ("StringData.txt");

---

Maximum marks: 4

### 3 Endurkvæmt

Skoðið eftirfarandi forrit. Hvað prentast út?

```
public class Leyndo {  
    public static void leyndo(int n) {  
        if (n <= 0) return;  
        System.out.print(n);  
        leyndo(n - 2);  
        leyndo(n - 2);  
        System.out.print(n);  
    }  
    public static void main(String[] args) {  
        leyndo(6);  
    }  
}
```

Veldu eitt af eftirtöldu

- 24222266222242
- 24666644666642
- 64222244222246
- ekkert, endalaus endurkvæmni
- 62444422444426

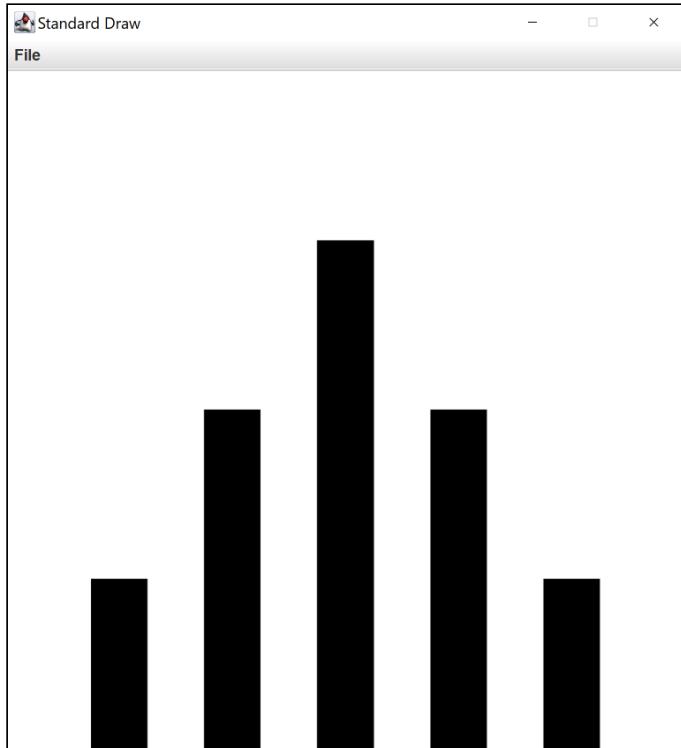
---

Maximum marks: 4

## 4 Súlurit StdStats

Ef eftirfarandi forritslínur eru notaðar til að teikna þetta súlurit, hvaða setning á að vera númer 2?

```
double[] fylki = { 1, 2, 3, 2, 1 };
// hvaða setning á að vera hér?
StdStats.plotBars(fylki);
```



Veldu eitt af eftirtöldu

- StdDraw.setXscale(0, StdStats.max(fylki) + 1);
- StdDraw.setYscale(0, StdStats.max(fylki) + 1);
- StdDraw.setScale(0, StdStats.max(fylki) \* 2 + 1);
- StdDraw.setYscale(0, StdStats.max(fylki));
- StdDraw.setXscale(0, StdStats.max(fylki));

---

Maximum marks: 4

## i Leiðbeiningar forritunarspurningar

Hér á eftir eru forritunarspurningar sem gilda **6 punkta** hver.

Gefinn er java ritill til að forrita í. Ritillinn er með setninganúmerum og sýnir java highlights með lit. Ritillinn hjálpar ykkur með réttan innadrátt á línum ef þið notið slaufusviga. Ritillinn athugar ekki hvort forritið er setningarlega (e. syntax) rétt og hann þýðir hvorki né keyrir forritið. Notið bestu venjur við forritun. Notið aðeins þá klasa sem farið hefur verið yfir í námsefninu.

**Þegar gefin er byrjun á forriti getið þið afritað hana í ritilinn.**

Þið þurfið ekki að lýsa forritinu með því að bæta við haus eða javaDoc nema sé beðið um það sérstaklega

## 5 Leita javaDoc og mynsturfar

Skrifaðu mynsturfar (e. signature) og javaDoc lýsingu fyrir static fallið **leitaOrd** í klasanum **Leita** sem tekur inn two strengi sem viðfangsbreytur (parameter), **s** og **t**, og skilar (return) hve oft **s** kemur fyrir í **t**.

Byrjunin á forritinu er svona. Afritaðu í ritilinn

```
public class Leita {
```

```
}
```

**Forritið hér**

---

Maximum marks: 6

## 6 Vara plús

Gefinn er eftirfarandi API fyrir óbreytanlegan (e. immutable) klasa **Vara**.

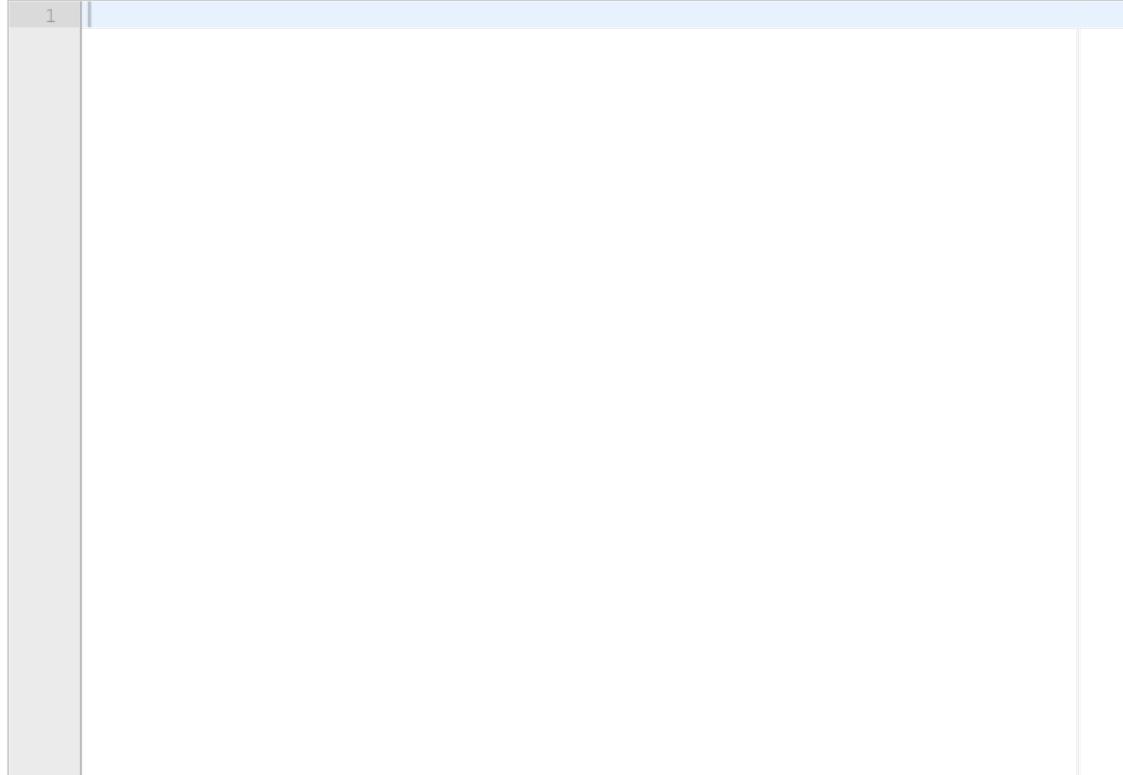
Skrifið public aðferðina **plus** í klasanum **Vara** sem leggur saman tvær vörur með því að leggja saman verð varanna og virðisaukaskatt varanna. Klasinn er óbreytanlegur (e. immutable).

```
public class Vara {  
    private final int verd;  
    private final int vsk;  
  
    public Vara(int v, int vsk) {...}  
  
    public Vara plus(Vara v) {...}
```

Byrjunin á forritinu er svona. Afritaðu í ritilinn

```
public class Vara {  
    // tilviksbreytur og smiður - þú þarf ekki að forrita smiðinn  
    public Vara plus (Vara v) {  
        // forritaðu hér  
    }  
}
```

Forritið hér



---

Maximum marks: 6

## 7 Skakbord - hreiðraðar lykkjur

Skrifaðu aðferðina **public static void teiknaBord(int n)** í klasanum **Skakbord** sem prentar út **n** sinnum **n** skákborð þar sem hvítir reitir eru " \* " (eitt bil \* eitt bil) og svartir reitir eru "---" (þrjú strik).

Ef n er 5 er úttakið svona

```
* --- * --- *
--- * --- * ---
* --- * --- *
--- * --- * ---
* --- * --- *
```

Byrjunin á forritinu er svona. Afritaðu í ritilinn

```
public class Skakbord {
    public static void teiknaBord(int n) {
        // forritaðu hér
    }
}
```

Forritið hér

---

Maximum marks: 6

## 8 Finna summu

Tvívítt fylki, **tolur**, af stærðinni NxN hefur heiltölur. Skrifið fallið **summa** sem fyrir stakið tolur[i][j], finnur summu talnanna í 8 nágranna stökum og skilar henni . Nágrannareitir tolur[i][j] eru reitir beint fyrir ofan, i-1, beint fyrir neðan stakið, i+1, til vinstri, j-1, til hægri j+1 og í hornunum [i-1][j-1], [i-1][j+1] o.s.frv. Reiknið með að i vísi ekki í fyrstu og síðustu röðina og j vísi ekki í fyrsta og síðasta dálkinn. Ef fylkið **tolur** lítur svona út skilar summa(1,1, tolur) tölunni 19. P.e.  $1+2+3+3+2+3+4+1=19$ .

**tolur** fylkið

1	2	3	4
1	2	3	4
4	3	2	1
1	1	1	1

Byrjunin á forritinu er svona. Afritaðu í ritilinn og ljúkið við forritið

```
public class Summa {
    public static int summa(int i, int j, int [][] tolur ) {
        // forritið hér
    }
}
```

Forritið hér

1	

Maximum marks: 6

## i Lengri forritunarspurningar

Hér á eftir eru forritunarspurningar sem gilda **12 punkta hver.**

Gefinn er java ritill til að forrita í. Ritillinn er með setninganúmerum og sýnir java highlights með lit. Ritillinn hjálpar ykkur með réttan innadrátt á línum ef þið notið slaufusviga. Ritillinn athugar ekki hvort forritið er setningarlega (e. syntax) rétt og hann þýðir hvorki né keyrir forritið. Notið bestu venjur við forritun. Notið aðeins þá klasa sem farið hefur verið yfir í námsefninu.

**Þegar gefin er byrjun á forriti skulið þið afritað hana í ritilinn.**

Þið þurfið ekki að lýsa forritinu með því að bæta við haus eða javaDoc nema sé beðið um það sérstaklega

## 9 Öryggistala

Til að athuga hvort kennitala er rétt er reiknuð svokölluð öryggistala út frá fæðingardegi einstaklings. Öryggistalan birtist í sæti 9 í kennitölunni.

T.d. ef tekin er kennitalan 1201603389 – þá er öryggistalan 8.

Hún er reiknuð á eftirfarandi hátt

stuðull	3	2	7	6	5	4	3	2
kennitala	1	2	0	1	6	0	3	3
margfeldi	3	4	0	6	30	0	9	6

Talnaröðin er margfölduð með 2-7 eins og hér er sýnt. Niðurstöðurnar eru svo lagðar saman og útkoman er 58. Þá er 11 deilt í 58 og fenginn afgangur sem er 3. Afgangurinn er dreginn frá 11, það er  $11-3=8$ . Öryggistalan er því 8.

Skrifið main-fall í klasanum **Kennitala** sem tekur inn streng af skipanalínu, reiknar út öryggistöluna og prentar út skilaboð um hvort kennitalan er rétt.

Vísbinding: Ascii heiltala fyrir bókstafinn 1 er 49, fyrir 2 er 50 o.s.frv.

Notkunardæmi

%java Kennitala 1201603389

Kennitala er rétt

%java Kennitala 1201603369

Kennitala er röng

Afritið eftirfarandi byrjun í ritilinn hér á eftir

```
public class Kennitala {
    public static void main(String[] args) {
        String kennitala = args[0];
        int[] studull = { 3, 2, 7, 6, 5, 4, 3, 2 };

        // forritið hér
    }
}
```

**Forritið hér**

1	
---	--

---

Maximum marks: 12

## 10 Deilanlegur - for lykkja

Skrifið fallið **runa** í klasanum **Deilanlegur** sem tekur inn sem viðfangsbreytur heiltölur **fjoldi**, **tala** og heiltölufylkið **gogn**, sem er af ótilgreindri stærð, og skilar streng. Finnið talnarunu úr **gogn** sem hefur, að *hámarki* **fjoldi** stök sem eru deilanleg með **tala** og búið til streng þannig að það sjáist númer hvað talan er í rununni og hve mörg stök eru í rununni. **runa** skilar þessum streng.

Notkunardæmi: Ef fjoldi er 2 og tala er 5 og gogn er 5 3 25 15 þá er strengurinn sem runa fallið skilar

1/2:5

2/2:25

Notkunardæmi: Ef fjoldi er 5 og tala er 5 og gogn er 5 3 25 15 30 þá er strengurinn sem runa fallið skilar

1/4:5

2/4:25

3/4:15

4/4:30

Afritið eftirfarandi byrjun í ritilinn hér á eftir

```
public class Deilanlegur {
    public static String runa(int fjoldi, int tala, int[] gogn) {
        // Forritaðu hér
    }
}
```

**Forritið hér**

---

Maximum marks: 12

## 11 Bil

Forritið tilviksbreytur, smið og aðferðir Bil klasans. Bil hlutur er talnabil frá min til max að báðum tölunum meðtöldum.

public class Bil		
	Bil (double min, double max)	bil frá min til max að báðum meðtöldum. Reiknið með að min<max
boolean	inniheldur (double x)	skilar true ef x er innan talnabilsins (báðar tölur meðtaldar)
boolean	skarast (Bil b)	skilar true ef b og þetta bil skarast (e. intersects)

Dæmi um skörun bila - ekki tæmandi

bilin [3.5, 4.5] og [2.5, 4.0] skarast

bilin [3.5, 4.5] og [2.5, 3.0] skarast ekki

Bil klasinn byrjar svona. Afritaðu í ritilinn.

```
public class Bil {
    // tilviksbreytur

    // smiður
    public Bil(double min, double max) {
        // forritaðu hér
    }
    public boolean inniheldur(double x) {
        // forritaðu hér
    }
    public boolean skarast(Bil b) {
        // forritaðu hér
    }
    // þarf ekki að forrita
    public static void main(String[] args) {}
}
```

**Forritið hér**

1	
---	--

---

Maximum marks: 12

## 12 Kolefnisspor

Forritið main-fall í klasanum Kolefnisspor til að reikna út og prenta meðalkolefnisspor ferða og heildarkolefnisspor einstaklings vegna ferða.

Ferðamátar og kolefnisspor þeirra eru í fylkjunum

**double [] spor; // dæmi um útprentað fylki er [1.0, 7.8, 1.1, 2.5]**

**String [] ferdamati; // dæmi um útprentað fylki [ganga, rafbíll, rafhlaupahjól, strætó]**

Dæmi - ganga hefur kolefnisspor 1.0 og rafbíll hefur kolefnisspor 7.8

Lesið ferðir einstaklings úr skránni **ferdir.txt**. Í hverri línu er ein ferð sem er lýst með runu af pörum, fyrra gildið í parinu er strengur sem segir til um ferðamátann og það seinna heiltala segir til um fjölda kílómetra sem einstaklingur ferðast með ferðamátanum. Athugið að allar tölur eru skálðaðar.

Dæmi um **ferdir.txt** - ferðamátarnir geta verið mismargir í hverri ferð. Í fyrstu ferðinni hefur einstaklingurinn ferðast með strætó í 5 km og svo á rafhlaupahjóli í 3 km. Í annarri ferðinni hefur einstaklingurinn keyrt 3 km í rafbíl.

strætó,5,rafhlaupahjól,3

rafbíll,3

strætó,10,rafhlaupahjól,2,ganga,3

Þið fáið aðferðina **private static int ixFerdamati(String s, String[] safn)** sem þið getið kallað á til að fá sæti ferðamáta **s** í fylkinu **safn**. Ef kallað er á fallið með **ixFerdamati ("strætó", ferdamati)** fæst heiltalan 3 sem hægt er að nota til að finna kolefnisspor strætó í fylkinu **spor**.

Dæmi um úttak gæti verið svona (úttak þarf ekki að vera sniðið (e. formatted)):

meðalkolefnisspor í 3 ferðum er 23.13

heildarkolefnisspor einstaklings var 69.40

Forritið byrjar svona. Afritið í ritilinn

```
public class Kolefnisspor {
```

```
    public static void main(String[] args) {
```

```
        double[ ] spor; // Gefið ykkur að spor sé upphafsstillt
```

```
        String[ ] ferdamati; // Gefið ykkur að ferdamati sé upphafsstillt
```

```
        In in = new In("ferdir.txt");
```

```
        // Forritaðu hér
```

```
}
```

**Forritið hér**

1	
---	--

---

Maximum marks: 12

## 13 Lagalisti

Gefinn er API fyrir klasann **Lag** sem inniheldur titil, flytjanda og hve oft lagið hefur verið spilað í vikunni. Aðferðir Lag klasans eru smiður, spila() sem hækkar fjölda spilana um einn og toString(). Þið þurfið ekki að útfæra Lag klasann.

Þið eigið að forrita klasann **Lagalisti** sem er klasi fyrir vinsældalista af stærð n.

Eftirfarandi skilyrði gilda um lagalista. Gætið þess að þau haldi eftir kall á hverja aðferð.

- Lagalisti er alltaf raðaður eftir vinsældum, þ.e. hve oft lagið hefur verið spilað
- Lagalistinn hefur alltaf n lög

Forritið Lagalisti klasann (tilviksbreytur og aðferðir og compare aðferðina) sem á að geta gert eftirfarandi:

- **Smiða** lagalista af stærð n úr skrá nafnASkra. Gefin er aðferðin

**public void lesaLog(String nafnASkra, Lag [] listi)** sem þið getið kallað á í smiðnum til að lesa lög úr skrá sem mynda upphafslistann. Forskilyrðið fyrir lesaLog er að **listi** fylkið rúmi n stök áður en kallað er á **lesaLog**.

- Bæta við lagi á listann. Ef lagið er óvinsælla en óvinsælasta lagið á listanum þá er laginu ekki bætt við. Annars er óvinsælasta lagið tekið af listanum og nýja laginu bætt við á listann

Afritið í ritilinn og forritið

```
public class Lagalisti {
    // tilviksbreytur

    // smiður
    public Lagalisti(String nafnASkra, int n) {
        // forritið hér
    }

    public void baetaVidLagi(Lag l) {
        // forritið hér
    }

    private class LagComparator implements Comparator<Lag> {
        public int compare(Lag o1, Lag o2) {
            // forritið hér
        }
    }

    private Comparator<Lag> getLagComparator() {
        // forritið hér
    }

    // Gefin aðferð - ekki forrita - bara nota
    private void lesaLog(String skra, Lag[] log){}
}

// ekki forrita main aðferð
public static void main(String[] args) {}

}
```

**Forritið hér**

1	
---	--

---

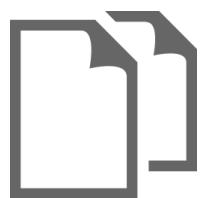
Maximum marks: 12





## Question 13

Attached



```
public class Lag {  
    private String titill;  
    private String flytjandi;  
    private int spilun;  
  
    public Lag(String t, String f, int s) {...}  
  
    public int getSpilun() {...}  
  
    public void spila() {...}  
  
    public String toString() {...}  
  
    public static void main(String[] args) {}  
}
```